REPRESENTING OPERATIONS PROCEDURES USING TEMPORAL DEPENDENCY NETWORKS

Kristina E. Fayyad and Lynne P. Cooper

Monitor & Control Technology Group
M/S 525-3660
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
kristina@isd.jpl.nasa.gov
lpcooper@ai.jpl.nasa.gov

ABSTRACT

DSN Link Monitor & Control (LMC) operations consist primarily of executing procedures to configure, calibrate, test, and operate a communications link between an interplanetary spacecraft and its mission control center. Currently, the LMC operators are responsible for integrating procedures into an end-to-end series of steps. The research presented in this paper is investigating new ways of specifying operations procedures that incorporate the insight of operations, engineering, and science personnel to improve mission operations. The paper describes the rationale for using Temporal Dependency Networks (TDNs) to represent the procedures, a description of how the data is acquired, and the knowledge engineering effort required to represent operations procedures. Results of operational tests of this concept, as implemented in the LMC Operator Assistant Prototype (LMCOA), are also presented.

Keywords: Automation, Knowledge Engineering, Operations

1. INTRODUCTION

DSN Link Monitor & Control (LMC) operations consist primarily of executing procedures to configure, calibrate, test, and operate a communications link between an interplanetary spacecraft and its mission control center [Ref. 1]. The procedures can be organized according to two taxonomies: those based on the individual subsystems which form the link (e.g. antenna, receiver, telemetry processor), and those specific to the spacecraft or science experiment. Currently, the LMC operators must manually integrate the individual subsystem and higher level spacecraft procedures into an end-to-end series of steps needed to support realtime operations. On-the-job experience provides the necessary background for determining any interdependencies between subsystems, for interpreting mission specific configuration and performance requirements for routine, often-performed operations, and for determining the fastest, most reliable means of getting the job done. Non-routine operations, however, often require operators to perform different types of operations and to interpret results in ways different than they normally would. For example, during an often performed pass (e.g. telemetry), the primary goal is to maximize the signal quality. Operators may adjust the various equipment in order to improve the signal-to-noise ratio (SNR). For a radio science type pass, performed much less frequently however, the configuration of the equipment must remain constant and the SNR may actually be part of the data collected.

The purpose of our current research is to investigate new ways of specifying operations procedures that enable automated operations, capture the insight and expertise of operations, engineering, and science personnel, and lead to more efficient and reliable operations. [Ref. 2] The goals for the representation are:

1. Extensible: it must be capable of representing the full spectrum of operations procedures.
2. Flexible: it must allow for the variations in operations procedures experienced between the different operations complexes, and under different circumstances.
3. Maintainable: because the station equipment and types of operations are constantly changing, it must be easy to update and maintain.
4. Verifiable.: the representation must be testable for accuracy.
5. Robust: it must provide the information necessary to:
   - identify problems
   - perform workarounds
   - interpret monitor data
   - tailor a general procedure to specific circumstances

6. User Natural: it must be readable and usable by both computer and human operator without translation.

2. APPROACH

Our approach is to use a temporal dependency network (TDN) to represent LMC operations procedures. A TDN is a directed graph which incorporates temporal and behavioral knowledge and also provides optional and conditional paths through the network. The directed graph (or, Petri Network) represents the steps required to perform an operation. Precedence relationships (step A has to happen before step B) are specified by the nodes and arcs of the network. The behavioral knowledge identifies sys[c,rn-slate dependencies in the form of pre- and post- conditions. Temporal knowledge consists of both absolute (e.g. Acquire the spacecraft at time 02:30:45) and relative (e.g. Perform step Y S minutes after step X) temporal constraints. Conditional branches in the network are those performed only under certain conditions. These are the IF (this condition) THEN (do/don't do that action). The conditionals are used primarily for error recovery. Optional paths are those which are not essential to the operation -- but may, for example, provide a higher level of confidence in the data if performed.

The TDN representation is discussed in more depth in a following section. In order Lo understand the representation it is necessary to first under'stand the basic building block of the TDN, directives.

2.1 Directives

The primary data unit of the TDN is a directive. A directive is a control message which is sent LO an individual subsystem in order to perform a specific function. The primary data fields are the intended subsystem, the control action, and any associated parameters. To support the. TDN mode.t of procedures, we use a more dc.tailed representation of the directives. Each directive is represented as an object containing the following information. An example of a directive definition is presented in Figure 1.

1. Directive syntax (subsystem, message name, required and optional parameters).
2. The function of the directive: what primary and side effects it has on the subsystem; what changes it causes in any devices or subassemblies.
3. Parameter definitions: any constraints on the, parameters and the support data used to determine parameter values,
4. Directive responses: the response messages sent from the subsystem to the LMC to acknowledge receipt of the directive. This is only a communications handshake and does not indicate that the directive was successfully executed.
5. Rejection notices: messages sent by the subsystem when the directive has failed to execute. (Includes syntax errors as well as realtime failures).
6. Monitor and event information: data that may be generated by the subsystem based on the actions of the directive. Specifics which parameters and user interface displays to monitor to confirm that the directive has Successfully executed.
7. Pre.-condi[ions: what state. must the system be in before this directive can be sent.
8, Post-conditions: which state the system is in when the directive has successfully executed.

INSERT FIGURE 1 EXAMPLE DIRECTIVE

The information in the directive definitions is stored in a knowledge base. Of the above listed types of information, only a subset, dealing primarily with syntax and gene.r-al responses, is available in the. DSN documentation. Much of the information is available only through the operations personnel (monitor information, pre- & post- conditions) or the engineers (Side effects, pre- & post- conditions).

2.2 Temporal Dependency Network

A TDN is a complex object which encodes the information necessary to perform a specific operational task. As described earlier, the primary representation of the TDN is an augmented directed graph. In the graph, each arc represents a strict precedence relationship, each node a sequence of directives which perform a subset of the overall function. The network explicitly specifies the precedence relationships between nodes, any potential parallelism, and rules for recovering from global faults. The nodes, or blocks, consist of the directives, temporal constraints, pre- and post-conditions, and local recovery information should the block fail. An example block is given in Figure 2.

INSERT FIGURE 2 EXAMPLE BLOCK

Once the directives necessary to perform the given operation and any pre- and post- conditions specific to the type of pass are known, designing a TDN becomes an exercise in assigning directives to the correct block. Preconditions specify device states that must be true. before the directive can execute.

Postconditions specify the expected device states after the directive has successfully executed. Precedence relationships in the TDN are formed by ensuring that the actions required to satisfy a directive precondition occur and are verified before it executes. SO, if two directives are in sequence because one depends on the successful completion of the other, these directives will be placed in separate blocks and a precedence relationship formed between them.

Directive preconditions are pushed up to the block level, so that before the block begins executing its first directive, all preconditions of all directives in that block must be satisfied. In some cases, this check is redundant because completion of the previous block is dependent upon satisfying a postcondition which satisfies the precondition of the next Mock. We have. designed the TDN in this way for two reasons: 1) the block may, at some point be moved to a different location in the TDN, and 2) if a device fails between the end of the first block and the start of the second, we have a way 10 detect the failure before proceeding.

'1 'he"1 'DN is the general representation of an operational task. An instance of the TDN is created from the general representation and parameterized for the speci fic pass being performed. From this perspective, the, TDN acts as a template for operations, and individual parameters (time, frequency, file names) are filled in at execution time to perform operations. A sample of a general TDN, shown at the block level, is given in Figure 3.

INSERT FIGURE 3 EXAMPLE TDN -- Block level

3. KNOWLEDGE ENGINEERING

'1'here are two knowledge bases required for the TDN. The first is the TDN itself, which is a high-level procedural representation. The second is the Directive Dictionary which contains the definitions for all of the directives used in our test TDN. An overview of the, knowledge engineering activities used to create both k nowledge bases is gi ven in the. following sections. For the purposes of this paper, we define knowledge engineering to be the. process of acquiring, representing, testing, and validating the knowledge in a knowledge base. In our Case, the knowledge engineering process was tightly coupled to the design process, and the results of our knowledge engineering efforts influenced overall system design of the. LMCOA Prototype.

Common to both knowledge engineering efforts was the need to keep detailed information as to the sources of information. The process of gathering knowledge is iterative and the sources may be referred back to mul tiple times, For example., the information from different personnel and documentation may be contradictory, so it may be necessary to refer back to each source to determ inc why. A significant part of our knowledge base development was dedicated to documenting the source of information, the date., and subsequent changes,

3.1 TDN Knowledge 1 ;ngineering

The 1'1)N concept for procedure representation is being tested by creating a TDN to support precalibration and data collection for a Very Long Baseline Interferometry (VI ,BI) Delta Differential One-Way Ranging (DDOR) pass at the Goldstone 70 Meter Antenna. Our first step in representing the VLBI DDOR was to create a baseline sequence of dire.ctives and other actions that the operators execute. While acquiring this information from various operators and engineers, it became clear that there is no single sequence that defines this activity. Since several subsystems are necessary and a significant part of their setup is independent of the other subsystems, the representat ion had to incorporate the network features of the TDN from the start to represent this inherent paral lelism. In addition, other variations in the procedures would arise when problems were detected and steps taken to resolve them during actual Operations.

The primary source for information concerning the TDN was through people. In order to effectively interview and acquire knowledge from the operations personnel, we needed a basis for discussion. We began by pulling all of the directives from a log of a sample pass. We created a draft TDN, in the form of individual directives (no blocks at this point) using post-it notes on (a 22 ft. length of) butcher paper[1]. The operations personnel marked up the'1 'DN and moved around the different directives. All comments resulting in changes or additions to the TDN included the source for the information, This proved extremely useful, especially when contradictions in input arose,

The contradictions themselves were a form of knowledge. They represented a wide variety of causes including: 1) A difference between what the scientists wanted and what operations personnel thought they wanted; 2) Procedure data that was valid only as a work around until a system bug was

[1]To our knowledge, this is the first time a complete cad-to-end sequence for an operational proced urc has ever been fully documented.

corrected; 3) differences in how the engineers designed the subsystems and how the operators actually use them; 4) Errors in the documentation. The TDN knowledge is still in the process of being validated by review with operations personnel and both laboratory and on-site testing.

3.2 Directive Dictionary Knowledge Engineering

The directive dictionary contains the directives for the subsystems necessary to perform a VLBI DDOR: antenna, microwave, receiver/exciter, precision power monitor, metric data assembly, and VLBI/signal processor. The knowledge that is collected for a single directive consists of: directive syntax and parameters, actions of the directive, directive responses, event notice messages and monitor data which may be generated based on the actions of the directive., and rejection notices, In addition, it is necessary to find out what subsystem devices these directives affect and are being referred to in these various messages. The original source for the majority of the directive information was the operations manuals for the individual subsystems. However, the majority of the effort expended in generating the Directive Dictionary was spent filling in the gaps and correcting inconsistencies in the manuals by using other sources, For example, LMC operator logs were used to identify directive responses and some of the event notice messages that can be expected.

identifying the preconditions necessary before a directive can successfully execute and the post-conditions after it executes required the collection and cross-correlation of many pieces of information. Each subsystem may consist of several devices and subassemblies. In many cases, the directive is sent to the. subsystem controller which in turn sends a control message to the appropriate device. In these circumstances, the assistance of the engineering staff was essential to determining what the dc.sired effect of the directive, really was and what information to monitor to determine that it had, in fact, executed. Some precondition information was available from the. manuals in the. form of device states which must be true., or other directives which must have been successfully completed before executing a specific directive. The actions of the pre-ceding directive can be translated into a pre- or post- condition. Precondition information was also gathered from evaluating event notice message data which specified error conditions when a directive failed.

3.3 Knowledge Engineering Summary

The knowledge engineering effort for the VLBI DDOR procedure currently consists of approximately 110 directives and 45 blocks. The sources used for knowledge engineering and the types of information obtained from those sources are listed in Table 1. The main thrust of the knowledge engineering effort has taken place over the past year. During that time, the DSN operational stations have undergone several modifications and updates to operational software. A major challenge in the knowledge engineering effort has been not just to capture the knowledge, but to aiso keep it current, For example, during our effort, the recommended precalibration procedure outlined by the antenna operations engineer has changed over 8 times.

4. STATUS

The knowledge engineering and TDN development activities have been performed in support of the LMC Operator Assistant Prototype (LMCOA) advanced development effort, In September 1992, the LMCOA had reached the point where its functionality and design [Ref. 3, 4] could be tested in an operational situation. After successful compatibility and functionality testing, we began testing the accuracy and completeness of the knowledge bases necessary to demonstrate automated[2] operations. These tests have included evaluation of the information in the Directive Dictionary in addition to end-to-end procedure testing of the TDN. in preliminary tests, the TDN has proven successful in meeting the goals outlined in Section 1 of this paper. A full scale demonstration of the LMCOA is scheduled for December 1992. The LMCOA is implemented on a NeXT Workstation using Objective C, Interface Builder, and CLIPS.

2 To be precise, it will be semi-automated operations. "The microwave subsystem requires manual configuration and there are actions, such as a safety page to warn people that the antenna wiii be moving, which the operator is required to perform,

Knowledge Source	Knowledge Obtained
Operations Manuals and operations Procedures	Directive Syntax Directive Responses Procedural Overview
logs	Sequence of directives Sample responses Sample monitor data and event messages Sample anomalies
1 ,MC Operators	Pre- & Post- conditions Support data Sequence of directives What they actually monitor, displays used System quirks Anomaly recovery actions False alarms and "noisy" feedback
Operations Engineers	Desired operations sequences Pre- & Post-conditions Preventative actions Subsystem quirks
Design Engineers	Device and subassembly reactions Side effects
Scientists	Desired sequence of "actions" (the events that should lake place, rather than the specific directives) What's important to the success of the experiment Rules for improving data quality

Table 1: Summary of Knowledge Sources

5. ACKNOWLEDGMENTS

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The other members of the LMCOA team arc Lorrine Lee, Randy Hill, Sanguan Chow, Kathy Sturdevant, and Juan Urista. We would like to acknowledge the contributions of the many operations and engineering personnel at JP1, and the Goldstone Deep Space Communicant ions Complex.

6, REFERENCES

1. Cooper, Lynne P., Rajiv Desai and Elmain Martinez, "Operator Assistant to Support Deep Space Net work Link Monitor and Control," SOAR Symposium, Houston, TX, 1991.

2. Cooper, Lynne P., "Operations Automation Using Temporal Dependency Networks," Technology 2001, San Jose, CA, December 3-5,]99],

3. 1 Ii Ii, Randall W., Jr. and Lorrine F. Lee "Situation Management in the Link Monitor and Control Operator Assistant (LMCOA)", SpaceOPS 92: Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, CA, 1992.

4. Lee, Lorrine and Randall W. }1 ill, Jr., "Process Control and Recovery in the Link Monitor and Control Operator Assistant," SOAR Symposium, Houston, TX, 1992.

	Subsystem	Directive	Parameters
	AP	DLOAD	<set-id >
.g.	AP	DLOAD	CW

Actions: transmits a predict data set to the ACS

ources: doy-067-91.log, line 189
doy-115-90.log, line 56
doy-116-90.log, line 400
doy-313-91.log, line 95

recondit ions: predicts available at LMC

ostconditions: am-status predict-id ?set-id
acs-status predict-table filled
acs-status download-ok-p t (download success)

esponses: COMPLETED. PROCESSING DLOAD REQUEST

ejections: COMPLETED. INVALID PREDICT SET NAME

vent notice messages
PA 14: INTERPOLATING <set-id> SUB1
PA 14:PTS INTERP TO <time>
PA 14:DLOADING <set-id> SUB1 AT 80 SECS
PA 14:PTS DLOAD TO <time>
PA 14:APA DLOAD COMPLETE
PA 14:ACS CONFIRM DLOAD...
PA 14:ACS TIME <time>

Figure 1, Directive Example, AP DLOAD

Preconditions:
ACS finished resetting

Directive sequence:
AP CONN 14
AP ACS DELUT 299
AP ACS REQCORR
AP ACS SCAN OFF
AP ACM REQCNF DES
AP ACM REQCNF ACT

Postcondit ions:
ACS received connect
DELUT set to 299
Conscanning disabled

Figure 2. Block Example

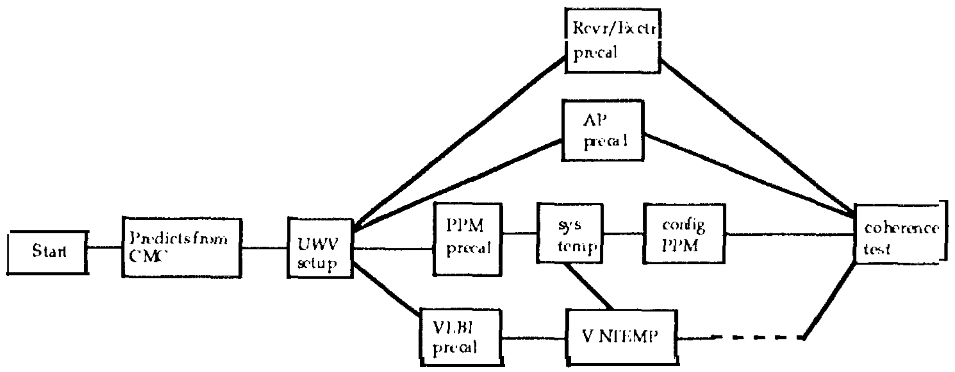

Figure 3. TDN Example, Block level